UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/621,207	07/15/2003	Rajeev Grover	200300624-1	1087

22879 7590 09/02/2009

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
3404 E. Harmony Road
Mail Stop 35
FORT COLLINS, CO 80528

EXAMINER
DAO, THUY CHAN

ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
09/02/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

JERRY.SHORMA@HP.COM
ipa.mail@hp.com
jessica.l.fusek@hp.com

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/621,207
Filing Date: July 15, 2003
Appellant(s): GROVER ET AL.

John P. Wagner, Jr.
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed May 27, 2009 appealing from the Office action mailed January 29, 2009.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

US Patent 6,339,832	Bowman-Amuah	04-2002

(9) Grounds of Rejection

The following grounds of rejection are applicable to the appealed claims:

❑ Claims 1-8, 10-15, and 17 are rejected under 35 U.S.C. 102(b) as being anticipated by Bowman-Amuah (US Patent No. 6,339,832).

Claim 1:

Bowman-Amuah discloses *an exception handling mechanism stored in one or more computer-readable storage devices, said exception handling mechanism comprising:*

an exception handler for recording exception information (e.g., FIG. 145, exception response table, col.262: 61 – col.263: 67; col.264: 45 – col.265: 33)

dependant on types of exceptions and programming tasks that encounter exceptions (e.g., FIG. 143, sheet 87, each type of exception A-C has each handling logic A-C, col.260: 54 - col.261: 37; col.263: 28-67; col.264: 9-44; col.265: 61 - col.266: 56)*; and*

"a recovery agent for taking an action upon an occurrence of an exception that occurred for a programming task" (e.g., FIG. 55, classes, sub-classes, and objects of batch jobs (*"programming tasks"*), block 5510 "Executing the logic of the batch job sub-classes associated with the identified object", emphasis added:

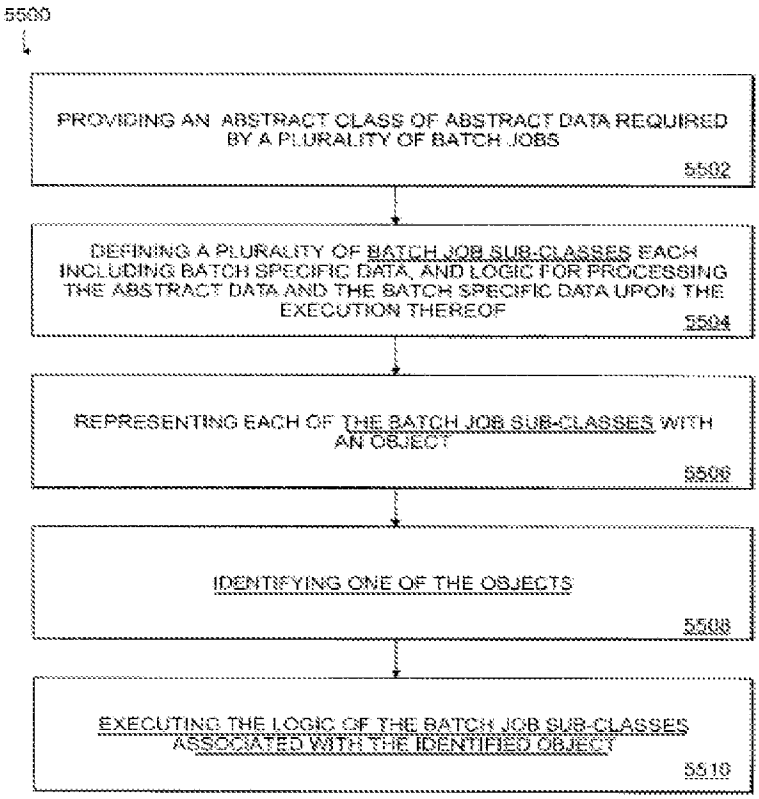

Fig. 55

"Once a batch job begins, it continues until it is complete or it encounters an error.

An architecture that supports batch jobs usually has certain characteristics. It must be able to support checkpoints and rollback, restart and recovery, error handling, logging, scheduling, and resource locking." (col.194, 7-12, emphasis added);

"In general, batch still has the following fundamental characteristics:

Scheduling--Services are required to manage the flow of processing within and between batch jobs, the interdependencies of applications and resources as well as to provide integration with checkpointing facilities.

> Restart /Recovery--Batch jobs must be designed with restartability in mind. This implies the need for a batch restart /recovery architecture used to automatically recover and re-start batch programs if they should fail during execution" (col.188: 34-44, restarting/recovering batch programs/jobs if they should fail/raise exception during execution, emphasis added); and
>
> "The examples have shown a single exception handler being used. In practice it is more likely that multiple will be used. For example, the exception handler on a server may have different requirements or constraints than a client, or one client may be GUI based and display pop-up error messages, where another client is a batch program that needs to send notification messages to Operations. This can be handled by creating multiple handlers or using the Strategy pattern to customize the behavior" (col.267: 36-45, exception handling for batch programs/jobs, emphasis added);

wherein the action is performed outside of a debugging operation (e.g., col.188: 34-4 as recited above, upon an occurrence of a failure/exception, the steps of restart / recovery of batch programs/jobs (*"programming tasks"*) are automatically performed/executed, which are completely separate/outside of a debugging operation (such as a debugging operation performed by a software developer/tester), emphasis added)*; and*

wherein the action to be taken upon the occurrence of the exception corresponds to a type of exception and a programming task (e.g., col.260: 54 – col.261: 37; col.263: 28-67; col.264: 9-44)*, and*

includes one or a combination of restarting the programming task, terminating the programming task (e.g., col.30: 12-18; col.92: 38-47; col.98: 24-34, restarting and/or terminating batch jobs),

resetting a system running the programming task, and disregarding the exception (e.g., col.108: 36-59; col.188: 23-51; col.260: 66 – col.261: 41),

wherein the exception handler and the recovery agent run on a first system (e.g., col.16: 14-33; FIG. 55, col.193: 42 - col.194: 34; FIG. 28, batch job exception handling and recovery components (first system), col.106: 65 – col.109: 34)

that operates autonomously and the first system is embedded in a second system (e.g., FIG. 10, col.31: 57 – col.32: 38, batch job exception handling and recovery components (first system) are embedded in main framework and/or base services (a second system); col.264: 45 – col.265: 33; col.294: 66 – col.295: 20).

Claim 2:

The rejection of claim 1 is incorporated. Bowman-Amuah discloses *the recorded exception information associated with an exception is associated with a signature for identifying the recorded exception information with its associated exception* (e.g., col.262: 61 – col.263: 67; col.260: 54 – col.261: 37).

Claim 3:

The rejection of claim 2 is incorporated. Bowman-Amuah discloses *the signature includes a version of a program running the programming task* (e.g., col.93: 24-50; col.193: 41 – col.194: 34).

Claim 4:

The rejection of claim 1 is incorporated. Bowman-Amuah discloses:

a plurality of sets of exception information for a plurality of exceptions is maintained in the system running the programming task (e.g., col.92: 38-47; col.98: 24-34; col.188: 36-51);

each set of exception information being associated with a signature for identifying that set of exception information (e.g., col.98: 24-34; col.193: 42 – col.194: 34).

Claim 5:

The rejection of claim 1 is incorporated. Bowman-Amuah discloses *the recorded exception information associated with an exception is associated with a signature for identifying the format of the exception information* (e.g., col.264: 9-44; col.265: 61 – col.266: 56).

Claim 6:

The rejection of claim 1 is incorporated. Bowman-Amuah discloses *the recorded exception information includes data related to a program stack, including data to reconstruct the program stack at time of exception* (e.g., col.16: 14-33; col.106: 65 – col.109: 34).

Claim 7:

The rejection of claim 1 is incorporated. Bowman-Amuah discloses *an analysis tool communicating via an interface with the system running the programming task, for identifying causes of the exception* (e.g., col.264: 9-44; col.265: 61 – col.266: 56).

Claim 8:

The rejection of claim 7 is incorporated. Bowman-Amuah discloses *the analysis tool uses a version to match the object code of a program running the programming task to the source code of the program* (e.g., col.31: 57 – col.32: 38; col.264: 45 – col.265: 33).

Claim 10:

Bowman-Amuah discloses *a processing system stored in one or more computer-readable storage devices, said processing system comprising:*

a first system (e.g., FIG. 10, col.31: 57 – col.32: 38; framework and/or base services; col.264: 45 – col.265: 33; FIG. 55, col.193: 42 – col.194: 34);

an autonomous second system embedded in the first system (e.g., FIG. 28, col.106: 65 – col.109: 34; batch job exception handling and recovery components (second system); col.264: 45 - col.265: 33)*;*

an exception handler running in the second system for recording exception information upon an occurrence of an exception in the second system (e.g., FIG. 145, col.262: 61 – col.263: 67; col.264: 45 – col.265: 33)*; and*

a recovery agent running on the second system (e.g., FIG. 55, col.193: 41 – col.194: 34, batch job recovery component; col.93: 24-50)*,*

for taking an action upon the occurrence of the exception based on the recorded exception information, wherein the action is performed outside of a debugging operation (e.g., FIG. 143, col.260: 54 – col.261: 37; FIG.10, Base Services with Batch jobs, col.31: 57 – col.32; 38; FIG. 28, Batch jobs exceptions are handled outside a debugging operation, col.106: 65 – col.109: 34)*;)*;

wherein the action corresponds to a type of exception that occurred in a programming task (e.g., col.264: 9-44; col.265: 61 – col.266: 56; FIG. 55, programming classes and sub-classes of batch jobs (programming task), col.193: 41 – col.194: 34; col.93: 24-50)*.*

Claim 11:

The rejection of claim is incorporated. Bowman-Amuah discloses *an analysis tool for receiving, via an interface, the recorded exception information from the second system and for identifying the cause of the exception* (e.g., FIG. 55, col.193: 41 – col.194: 34; col.92: 38-47; col.260: 54 – col.261: 37)*.*

Claim 12:

The rejection of claim 10 is incorporated. Bowman-Amuah discloses *the second system includes nonvolatile memory for storing exception information* (e.g., col.262: 61 – col.263: 67; col.106: 65 – col.109: 34; col.193: 42 – col.194: 34)*.*

Claim 13:

The rejection of claim 12 is incorporated. Bowman-Amuah discloses *the exception information stored in the non-volatile memory is compressed* (e.g., col.264: 45 – col.265: 33; col.31: 57 – col.32: 38).

Claim 14:

The rejection of claim 12 is incorporated. Bowman-Amuah discloses *the exception information stored in non-volatile memory includes a plurality of sets of exception information, each set being associated with an exception and a signature* (e.g., col.193: 41 – col.194: 34; col.93: 24-50; col.265: 61 – col.266: 56).

Claim 15:

Bowman-Amuah discloses *a computer-readable storage device having stored thereon a computing system comprising:*

an exception handler for recording exception information on non-volatile memory upon an occurrence of an exception (e.g., FIG. 145,col.262: 61 – col.263: 67; FIG.10, Base Services with Batch jobs, col.31: 57 – col.32; 38; FIG. 28, Batch jobs exceptions are handled outside a debugging operation, col.106: 65 – col.109: 34)*;*

a recovery agent for taking an action upon an occurrence of an exception that occurred for a programming task based on the recorded exception information (e.g., FIG. 55, classes, sub-classes, and objects of batch jobs (*"programming tasks"*):

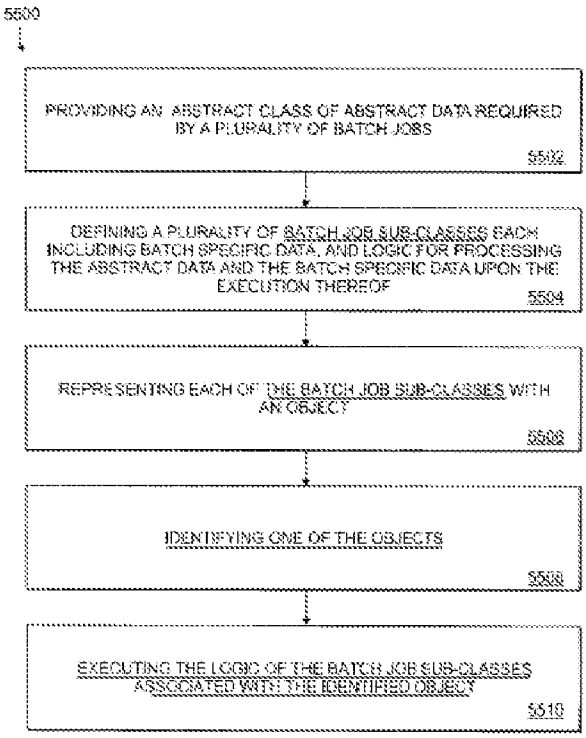

Fig. 55

"Once a batch job begins, it continues until it is complete or it encounters an error.

An architecture that supports batch jobs usually has certain characteristics. It must be able to support checkpoints and rollback, restart and recovery, error handling, logging, scheduling, and resource locking." (col.194, 7-12, emphasis added);

"In general, batch still has the following fundamental characteristics:

Scheduling--Services are required to manage the flow of processing within and between batch jobs, the interdependencies of applications and resources as well as to provide integration with checkpointing facilities.

Restart /Recovery--Batch jobs must be designed with restartability in mind. This implies the need for a batch restart

/recovery architecture used to automatically recover and re-start batch programs if they should fail during execution" (col.188: 34-44, restarting/recovering batch programs/jobs if they should fail/raise exception during execution, emphasis added); and

"The examples have shown a single exception handler being used. In practice it is more likely that multiple will be used. For example, the exception handler on a server may have different requirements or constraints than a client, or one client may be GUI based and display pop-up error messages, where another client is a batch program that needs to send notification messages to Operations. This can be handled by creating multiple handlers or using the Strategy pattern to customize the behavior" (col.267: 36-45, exception handling for batch programs/jobs, emphasis added);

wherein the action is performed outside of a debugging operation (e.g., col.188: 34-4 as recited above, upon an occurrence of a failure/exception, the steps of restart / recovery of batch programs/jobs (*"programming tasks"*) are automatically performed/executed, which are completely separate/outside of a debugging operation (such as a debugging operation performed by a software developer/tester), emphasis added)*; and*

an analysis tool for identifying the cause of the exception (e.g., col.263: 28-67; col.264: 9-44; col.265: 61 – col.266: 56)*;*

wherein the analysis tool receives the exception information from the nonvolatile memory via an interface interfacing a first system and a second system running the exception handler (e.g., col.106: 65 – col.109: 34; col.264: 45 – col.265: 33; col.193: 42 – col.194: 34) *and the recovery agent* (e.g., col.16: 14-33; FIG. 55, col.193: 42 - col.194: 34; FIG. 28, batch job exception handling and recovery components, col.106: 65 – col.109: 34)

wherein the second system (batch job exception handling and recovery components) *is embedded in a third system and the second system operates autonomously of other systems* (e.g., FIG. 10, col.31: 57 – col.32: 38; main framework, base services, batch job components (third system), batch job exception handling and recovery agents (second system) operates autonomously; col.106: 65 – col.109: 34).

Claim 17:

The rejection of claim 15 is incorporated. Bowman-Amuah discloses *the recorded exception information includes data related to a program stack* (e.g., col.10: 45 - col.12: 49; col.16: 14-33).

-o-o-o-

(10) Response to Argument

A) Bowman-Amuah does not disclose every claimed element (Brief, pp. 8-12):

Limitations at issue *"a recovery agent for taking an action upon an occurrence of an exception that occurred for a programming task, wherein the action is performed outside of a debugging operation"* (e.g., claim 1, lines 4-6).

As an initial matter, examiner notes that Appellants' arguments did not direct to the actual ground of rejection.

Moreover, the examiner's responses regarding these limitations at issue here, i.e., "batch job" versus "exception" was addressed in the Final office action mailed January 29, 2009, at pages 2-5, in which the Appellants have been completely silent and/or have not pointed out the supposed errors made by the examiner.

As clearly set forth in pages 2-5 of the previous Office action mailed January 29, 2009, the claimed limitation *"a programming task"* was equated with Bowman-Amuah's batch jobs and the claimed limitation *"an exception that occurred for a programming task"* was equated with Bowman-Amuah's batch jobs exceptions (but not "batch job" as contended/argued by the Appellants).

Bowman-Amuah explicitly teaches:

"a recovery agent for taking an action upon an occurrence of an exception that occurred for a programming task" (e.g., FIG. 55, classes, sub-classes, and objects of batch jobs (*"programming tasks"*), block 5510 "Executing the logic of the batch job sub-classes associated with the identified object", emphasis added:

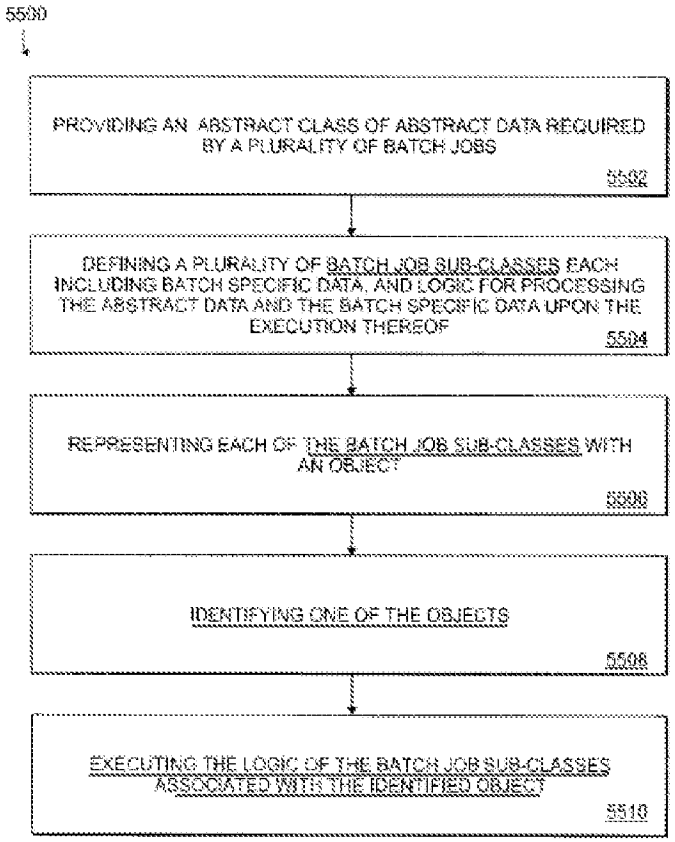

Fig. 55

"Once a batch job begins, it continues until it is complete or it encounters an error.

An architecture that supports batch jobs usually has certain characteristics. It must be able to support checkpoints and rollback, restart and recovery, error handling, logging, scheduling, and resource locking." (col.194, 7-12, emphasis added);

"In general, batch still has the following fundamental characteristics:

Scheduling--Services are required to manage the flow of processing within and between batch jobs, the interdependencies of applications and resources as well as to provide integration with checkpointing facilities.

> Restart /Recovery--Batch jobs must be designed with restartability in mind. This implies the need for a batch restart /recovery architecture used to automatically recover and re-start batch programs if they should fail during execution" (col.188: 34-44, restarting/recovering batch programs/jobs if they should fail during execution, i.e., raise exception during execution, emphasis added); and
>
> "The examples have shown a single exception handler being used. In practice it is more likely that multiple will be used. For example, the exception handler on a server may have different requirements or constraints than a client, or one client may be GUI based and display pop-up error messages, where another client is a batch program that needs to send notification messages to Operations. This can be handled by creating multiple handlers or using the Strategy pattern to customize the behavior" (col.267: 36-45, exception handling for batch programs/jobs, emphasis added);

"wherein the action is performed outside of a debugging operation" (e.g., col.143: 2-10, "Because components can be implemented in a variety of programming languages on a number of platforms, it is often necessary to have competencies in a number of technologies. For example, one client used Visual Basic, Smalltalk, C++, and COBOL for different layers of the system. The increasing number of technology combinations also increases the complexity associated with development activities such as testing, debugging, and configuration management." (i.e., typical software lifecycle has development stage/activities "such as testing, debugging", emphasis added);

col.188: 34-4, "Restart /Recovery--Batch jobs must be designed with restartability in mind. This implies the need for a batch restart /recovery architecture used to automatically recover and re-start batch programs if they should fail during execution" (i.e.,

upon an occurrence of a failure/exception "during execution", the steps of restart / recovery of batch programs/jobs (*"programming tasks"*) are automatically performed/executed during execution (during implementation stage/activities), which are completely separate/outside of a debugging operation (such as a debugging operation performed by a software developer/tester in the development stage/activities), emphasis added).

B) Bowman-Amuah does not disclose the claimed element as arranged by Appellants' claims (Brief, pp. 12-14):

The Appellants stated,

> "On further inspection of the cited portions, the instant Office Action cites portions of Bowman-Amuah that disclose different embodiments of the invention disclosed in Bowman-Amuah. While columns 260-265 discuss exceptions and exceptions handling, columns 106- 109 and 193-194 discuss batch jobs. As argued above, Appellants submit that exceptions and exceptions handling are not batch jobs…" (Brief, page 13, emphasis added).

As set forth in section A) above, examiner notes that Appellants' arguments did not direct to the actual ground of rejection.

As consistently set forth in pages 2-5 of the previous Office action mailed January 29, 2009 and in this examiner's answer, the claimed limitation *"a programming task"* was equated with Bowman-Amuah's batch jobs and the claimed limitation *"an exception that occurred for a programming task"* was equated with Bowman-Amuah's batch jobs exceptions (but not "batch job" as contended/argued by the Appellants).

C) Response to Arguments in the instant Office action (Brief, pp. 14-15):

The Appellants further stated,

> "Additionally, Appellants respectfully submit that "an exception that occurred for a programming task," as disclosed by Appellants' Claim 1 should not be equated with "Batch jobs exceptions," as is argued in the instant Office Action…" (Brief, page 14, last paragraph, original emphasis).

Examiner notes that Appellants' arguments fail to comply with 37 CFR 1.111(b). The reply by the Appellant or patent owner did not distinctly and specifically points out the supposed errors in the examiner's action and did not reply to every ground of objection and rejection in the prior Office action. A general allegation that the claims define a patentable invention ("an exception that occurred for a programming task," as disclosed by Appellants' Claim 1 should not be equated with "Batch jobs exceptions") without specifically pointing out how the language of the claims patentably distinguishes them from the references does not comply with the requirements of this section.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejection should be sustained.

Respectfully submitted,

/Thuy Dao/

Examiner, Art Unit 2192

Conferees:

/Tuan Q. Dam/

Supervisory Patent Examiner, Art Unit 2192

/Lewis A. Bullock, Jr. /

Supervisory Patent Examiner, Art Unit 2193